Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for controlling access to data handled by references in a system for executing programs (including processes and tasks), ~~characterized in that~~ wherein upon executing a program, it comprises the following steps:

- having the system store the whole of the references which the program obtains by means considered as licit;
- before any operation intended to be forbidden, if it deals with values which are not licit references, having the system check that these values are among the licit references which have been stored for this program, and acceptance or rejection of the operation accordingly.

2. (Currently Amended) The method according to claim 1, ~~characterized in that~~ wherein the references are pointers and/or handles.

3. (Currently Amended) The method according to claim 1, ~~characterized in that~~ wherein the licit means for a program in order to obtain reference values comprise at least one of the following operations:

- reading a variable or a datum belonging to the system or to another program,
- writing into a variable or datum of said program by the system or by another program,
- receiving arguments upon calling a routine of said program by the system or by another program,
- utilization of the return value from the call by said program of a routine belonging to the system or to another program,
- having said program catch up a raised exception during execution of a routine belonging to the system or to another program,
- receiving by said program an interruption or a valuated signal.

4. (Currently Amended) The method according to claim 1, ~~characterized in that~~ wherein:

- the system comprises a mechanism which determines whether a given value is a valid reference, and/or
- the stored licit references are limited to the sole references on data considered as sensitive for the system, and/or
- said checks check that the values are among the sensitive licit references which were stored for this program or else which are references determined as valid and dealing with data which are not sensitive.

5. (Currently Amended) The method according to claim 4, ~~characterized in that~~ wherein the system comprises a firewall which forbids certain operations by certain programs on certain referenced data, ~~the~~ data considered as being sensitive for the system being those for which the operations are not forbidden by the firewall.

6. (Currently Amended) The method according to claim 5, ~~characterized in that~~ wherein the firewall forbids certain operations by a program on data belonging to other programs, except on those declared as shareable.

7. (Currently Amended) The method according to claim 6, ~~characterized in that~~ wherein the system is based on a Java Card virtual machine and ~~in that~~ wherein:

- a program consists of the whole of the code which is found in a "Java Card package";
- the firewall is that of the Java Card Runtime Environment (JCRE);
- the data declared as shareable (and therefore sensitive) are objects which are instances of classes which implement the "Javacard.framework.Shareable" interface as well as, possibly, the objects with public use of the system: global arrays and Entry Point Objects of JCRE.

8. (Currently Amended) The method according to claim 7, ~~characterized in that~~ wherein the system stores in ~~the~~ sets of sensitive licit references associated with

a package all the references which appear in the following cases:

- receiving arguments of "Javacard.framework.Shareable" type when a method of said package is called by another package or by the system,
- "Javacard.framework.Shareable" type return value when said package calls a method from another package or from the system (including ~~the~~ a "getAppletSharreableInterfaceObject" method of "Javacard.framework.JCSystem package"),
- reading a public static field of "Javacard.framework.Shareable" type in another package or in the system,
- catching up an instance object of a class from (inheriting from) "java.lang.Throwable" and implementing "Javacard.framework.Shareable".

9. (Currently Amended) The method according to ~~any of claims 1 and 4,~~ claim 1, ~~characterized in that~~ wherein the whole of the licit ~~(or sensitive licit)~~ stored references is represented by a table.

10. (Currently Amended) The method according to ~~any of claims 1 and 4~~ claim 1, ~~characterized in that~~ wherein the set of the licit ~~(or sensitive licit)~~ stored references is emptied, by means of a possibly conservative garbage collector, of references which have become inactive.

11. (Currently Amended) The method according to ~~any of claims 1 and 4~~ claim 1, ~~characterized in that~~ wherein:

- the references are represented in the system by handles and tables of pointers ~~(or of references)~~,
- some of said tables are possibly reserved for licit ~~(or sensitive licit)~~ references,
- the sets of licit ~~(or sensitive licit)~~ stored references are represented by vectors ~~(or matrices)~~ of bits associated with some of the tables of pointers ~~(or references)~~, where a bit has a given index which represents the presence or the absence of the corresponding reference in said sets,
- said vectors of bits are possibly hollow and represented by means of a sequence of indexes or

lengths corresponding to the extents of bits positioned in the same way.